

**Yee &
Associates, P.C.**

4100 Alpha Road
Suite 1100
Dallas, Texas 75244

Main No. (972) 385-8777
Facsimile (972) 385-7766

**RECEIVED
CENTRAL FAX CENTER**

MAR 23 2005

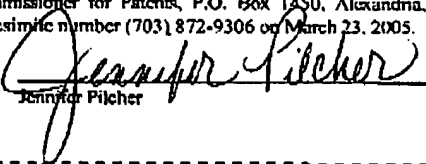
Facsimile Cover Sheet

To: Commissioner for Patents for Examiner Lillian Vo Group Art Unit 2127	Facsimile No.: 703/872-9306
From: Jennifer Pilcher Legal Assistant to Wayne Bailey	No. of Pages Including Cover Sheet: 29
Message: Enclosed herewith: <ul style="list-style-type: none">• Transmittal Document; and• Appeal Brief.	
Re: Application No. 09/690,457 Attorney Docket No: AUS9-2000-0370-US1	
Date: Wednesday, March 23, 2005	
Please contact us at (972) 385-8777 if you do not receive all pages indicated above or experience any difficulty in receiving this facsimile.	<i>This Facsimile is intended only for the use of the addressee and, if the addressee is a client or their agent, contains privileged and confidential information. If you are not the intended recipient of this facsimile, you have received this facsimile inadvertently and in error. Any review, dissemination, distribution, or copying is strictly prohibited. If you received this facsimile in error, please notify us by telephone and return the facsimile to us immediately.</i>

**PLEASE CONFIRM RECEIPT OF THIS TRANSMISSION BY
FAXING A CONFIRMATION TO 972-385-7766.**

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of: **Ku et al.**Serial No.: **09/690,457**Filed: **October 19, 2000**For: **Monitoring Modifications to
Environment Variables**§
§
§
§
§
§Group Art Unit: **2127**Examiner: **Vo, Lilian**Attorney Docket No.: **AUS9-2000-0370-US1****35525**PATENT TRADEMARK OFFICE
CUSTOMER NUMBER

<p>Certificate of Transmission Under 37 C.F.R. § 1.8(a)</p> <p>I hereby certify this correspondence is being transmitted via facsimile to the Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450, facsimile number (703) 872-9306 on March 23, 2005.</p> <p>By:  Jennifer Filcher</p>
--

TRANSMITTAL DOCUMENTCommissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

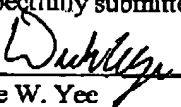
Sir:

ENCLOSED HEREWITH:

- Appeal Brief (37 C.F.R. 41.37).

A fee of \$500.00 is required for filing an Appeal Brief. Please charge this fee to IBM Corporation Deposit Account No. 09-0447. No additional fees are believed to be necessary. If, however, any additional fees are required, I authorize the Commissioner to charge these fees which may be required to IBM Corporation Deposit Account No. 09-0447. No extension of time is believed to be necessary. If, however, an extension of time is required, the extension is requested, and I authorize the Commissioner to charge any fees for this extension to IBM Corporation Deposit Account No. 09-0447.

Respectfully submitted,


Duke W. Yee
Registration No. 34,285
YEE & ASSOCIATES, P.C.
P.O. Box 802333
Dallas, Texas 75380
(972) 385-8777

ATTORNEY FOR APPLICANTS

Docket No. AUS9-2000-0370-US1

PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

RECEIVED
CENTRAL FAX CENTER

MAR 23 2005

In re application of: Ku et al.

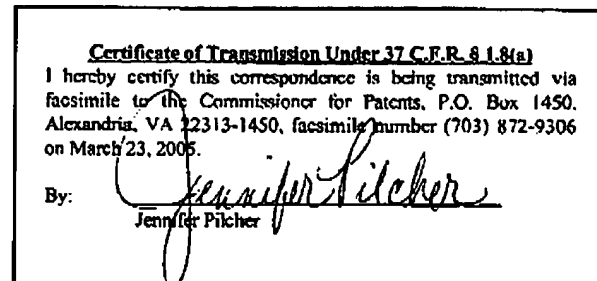
Serial No.: 09/690,457

Filed: October 19, 2000

For: Monitoring Modifications to
Environment Variables§
§
§
§
§
§
§

Group Art Unit: 2127

Examiner: Vo, Lilian

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

APPEAL BRIEF (37 C.F.R. 41.37)

This brief is in furtherance of the Notice of Appeal, filed in this case on January 24, 2005.

The fees required under § 41.20(B)(2), and any required petition for extension of time for filing this
brief and fees therefore, are dealt with in the accompanying TRANSMITTAL OF APPEAL BRIEF.(Appeal Brief Page 1 of 27)
Ku et al. - 09/690,457

REAL PARTY IN INTEREST

The real party in interest in this appeal is the following party: International Business Machines Corporation.

RELATED APPEALS AND INTERFERENCES

With respect to other appeals or interferences that will directly affect, or be directly affected by, or have a bearing on the Board's decision in the pending appeal, there are no such appeals or interferences.

STATUS OF CLAIMS**A. TOTAL NUMBER OF CLAIMS IN APPLICATION**

Claims in the application are: 1, 6, 7, 9, 10, 15-17 and 22-32

B. STATUS OF ALL THE CLAIMS IN APPLICATION

1. Claims canceled: 2-5, 8, 11-14 and 18-21
2. Claims withdrawn from consideration but not canceled: None
3. Claims pending: 1, 6, 7, 9, 10, 15-17 and 22-32
4. Claims allowed: None
5. Claims rejected: 1, 6, 7, 9, 10, 15-17 and 22-32
6. Claims objected to: none

C. CLAIMS ON APPEAL

The claims on appeal are: 1, 6, 7, 9, 10, 15-17 and 22-32

STATUS OF AMENDMENTS

No amendment after final was filed for this case.

SUMMARY OF CLAIMED SUBJECT MATTER

A. CLAIM 1 - INDEPENDENT

A method for correcting a path sequence of an environment variable in a data processing system, *the path sequence specifying an order for searching directories for locating executable code within the data processing system*. The method includes steps of (i) monitoring the data processing system for a change effecting the path sequence of the environment variable, where the environment variable is enabled and being used by the data processing system to specify the order for searching the directories within the data processing system, (ii) determining whether any duplicate files exist in any of the directories identified by the path sequence, and (iii) altering the path sequence of the environment variable to ensure that a proper file is found and used when selected by one of a user and a running application program. The above method is described at least at Specification page 7, lines 17-28 and page 9, line 8 – page 10, line 22 with reference to Figures 2 and 4.

B. CLAIM 9 - INDEPENDENT

A method for *correcting modifications that have been made to an environment variable* during installation of software in a data processing system. The method includes steps of (i) installing the software on the data processing system for subsequent execution by the data processing system, (ii) detecting that an environment variable has been modified during the installing step, (iii) responsive to the detecting step, determining if duplicate files exist in a path sequence of the modified environment variable, (iv) responsive to a determination that duplicate files exist in a path sequence of the environment variable, prompting a user to select a correct one of the duplicate files, and (v) removing all incorrect ones of the duplicate files from the path sequence of the environment variable. The above method is described at least at Specification page 7, lines 17-28 and page 9, line 8 – page 10, line 22 with reference to Figures 2 and 4.

C. CLAIM 10 – INDEPENDENT

Claim 10 is a computer program product claim with functionality as described above with respect to Claim 1.

D. CLAIM 17 - INDEPENDENT

Claim 17 is an apparatus claim having means for elements for performing the functionality as described above with respect to Claim 1.

E. CLAIM 24 - INDEPENDENT

A method for managing environment variables in a data processing system, with an environment variable manager that is automatically invoked to correct a path sequence. The method includes steps of (i) automatically invoking an environment variable manager upon occurrence of at least one of occurring events: a) a directory is deleted, b) a product is uninstalled on the data processing system, and c) a given environment variable is manually modified by a user; (ii) determining, by the environment variable manager, if any occurring event a), b) or c) causes a modification to an affected path sequence of any presently active environment variable, the path sequence specifying an order for searching directories for locating executable code within the data processing system; and (iii) automatically correcting the affected path sequence if it is determined that the occurring event causes the modification. The above method is described at least at Specification page 7, line 17 – page 10, line 22 with reference to Figures 2-4.

F. CLAIM 25 - INDEPENDENT

A method for managing environment variables in a data processing system, with an environment variable manager that is automatically invoked to provide a display for user correction of a path sequence based upon a determination of whether duplicate files exist in the directories specified by the path sequence of the environment variable. The method includes steps of (i) automatically invoking an environment variable manager whenever a path sequence for a presently active environment variable is modified in the data processing system, wherein the path sequence specifies an order for searching directories for locating executable code within the data processing system; (ii) determining, by the environment variable manager, if duplicate files exist in the directories specified by the path sequence of the environment variable; and (iii) enabling a display of each environment variable determined to have duplicate files in the directories specified by the path sequence to a user for correction. The above method is described at least at Specification page 7, line 17 – page 10, line 22 with reference to Figures 2-4.

G. CLAIM 26 – INDEPENDENT

A method for managing environment variables in a data processing system, and is specifically directed to *automatic or manual deletion of a directory* specified by a path sequence of an environment variable. The method includes steps of (i) determining if a directory, specified by a path sequence of any environment variable, is deleted, wherein the path sequence specifies an order for searching directories for locating executable code within the data processing system; and (ii) enabling at least one of the following: a) an automatic deletion of the directory from the path sequence of the environment variable, and b) a display of an interface to inform the user to delete the directory from the path sequence of any affected environment variables. The above method is described at least at Specification page 7, line 29 – page 9, line 7 with reference to flow-chart depicted in Figure 3.

H. CLAIM 27 - INDEPENDENT

Claim 27 is an apparatus claim having means for elements for performing the functionality as described above with respect to Claim 24.

I. CLAIM 28 - INDEPENDENT

Claim 28 is an apparatus claim having means for elements for performing the functionality as described above with respect to Claim 25.

J. CLAIM 29 - INDEPENDENT

Claim 29 is an apparatus claim having means for elements for performing the functionality as described above with respect to Claim 26.

K. CLAIM 30 - INDEPENDENT

Claim 30 is a computer program product claim with functionality as described above with respect to Claim 24.

L. CLAIM 31 - INDEPENDENT

Claim 31 is a computer program product claim with functionality as described above with respect to Claim 25.

M. CLAIM 32 - INDEPENDENT

Claim 32 is a computer program product claim with functionality as described above with respect to Claim 26.

GROUND OF REJECTION TO BE REVIEWED ON APPEAL

A. GROUND OF REJECTION 1 (Claims 1, 6, 10, 15, 17, 22, 24, 26, 27, 29, 30 and 32)

Claims 1, 6, 10, 15, 17, 22, 24, 26, 27, 29, 30 and 32 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Forbes et al. (U.S. Patent Publication No. 2002/0144248) (hereinafter "Forbes").

B. GROUND OF REJECTION 2 (Claims 7, 9, 16, 23, 25, 28 and 31)

Claims 7, 9, 16, 23, 25, 28 and 31 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Forbes et al. (U.S. Patent Publication No. 2002/0144248) (hereinafter "Forbes") as applied to claim 1 above, in view of Hove et al. (U.S. Patent No. 6,564,369) (hereinafter "Hove").

ARGUMENT

The claims do not stand or fall together, as described below in the separate claim arguments A1-A3 and B1-B3.

A. GROUND OF REJECTION 1 (Claims 1, 6, 10, 15, 17, 22, 24, 26, 27, 29, 30 and 32)

A.1. Claims 1, 6, 10, 15, 17 and 22

Appellants urge that the cited Forbes reference does not teach or suggest any type of *directory search order*, and therefore it necessarily does not teach or suggest any method for correcting a path sequence that specifies an order for searching directories for locating executable code within a data processing system. The cited Forbes reference teaches a software package manager for installing new software packages. An included manifest file includes install, execution, and uninstall information, and information to assist in identifying any software dependencies required for installation. The reference also describes a 'code store data structure' that is maintained on the computer for tracking what programs are installed on the computer and specifically where they are located. During installation of a software program, the software package manager can scan this code store data structure to determine if a component to be installed already exists on the computer, and update the code store data structure with the specific location of the later version of the component. When a user requests execution of a program, the code store data structure is queried to locate the appropriate components for use as explicitly specified in the code store data structure. *The code store data structure (which is clearly shown in Forbes Figure 4) does not provide any type of 'directory search' order, but rather describes installed programs and their explicit location* (see Forbes paragraphs 0054, 0064, 0073, 0080 and 0091, Figure 4, element 407, for example). Importantly, because the code store data structure explicitly saves the name of the directory of where a program has been installed on a computer system, there would have been no motivation to modify the teachings contained therein to include a directory search order, as searching of directories to locate a program file is not required by Forbes since the program location is explicitly stored in the code store data structure. Thus, it is urged that there is no teaching or suggestion by Forbes of any method for correcting a path sequence, where the *path sequence specifies an order for searching directories*, nor is there any motivation to modify such teachings in accordance with the claimed invention. Thus, it is

urged that Claim 1 has been erroneously rejected.

In addition, there is at least one additional specific claimed step not taught or suggested by the single reference being used in the 35 USC 103 rejection of Claim 1. The Examiner asserts that the missing claimed step of "responsive to determining that duplicate files do exist, altering the path sequence of the environment variable to ensure that a proper file is found and used when selected by one of a user and a running application program" is obvious, without providing any substantiating evidence. The Examiner merely argues that this claimed step would have been obvious in order for the Forbes' system to function correctly with proper files selected during run after undesired components have been removed. Appellants urge two-fold error in such assertion.

First, in rejecting claims under 35 U.S.C. Section 103, the examiner bears the initial burden of presenting a *prima facie* case of obviousness. *In re Oetiker*, 977 F.2d 1443, 1445, 24 USPQ2d 1443, 1444 (Fed. Cir. 1992). Only if that burden is met, does the burden of coming forward with evidence or argument shift to the applicant. *Id.* To establish *prima facie* obviousness of a claimed invention, all of the claim limitations must be taught or suggested by the prior art. MPEP 2143.03 (emphasis added by Appellants). *See also, In re Royka*, 490 F.2d 580 (C.C.P.A. 1974). The Examiner's mere assertion that a missing claimed feature would have been obvious does not meet this all-elements teaching/suggestion requirement. In the absence of a proper *prima facie* case of obviousness, an applicant who complies with the other statutory requirements is entitled to a patent. *See In re Oetiker*, 977 F.2d 1443, 1445, 24 USPQ2d 1443, 1444 (Fed. Cir. 1992), and thus Appellants urge that the Board reverse such rejection and allow these claims as the Examiner has failed to properly establish a *prima facie* showing of obviousness with respect to Claim 1.

Secondly, if undesirable elements have been removed (as described by the Examiner in their subjective reasoning of why this missing claimed step would have been obvious), Appellants urge that there would not necessarily be any reason to update any environment variable the specifies a directory search order as there would be no conflicts to be concerned about. Restated, since Forbes teaches that the actual location of the files is updated, there would be no reason to also update an environment variable that specifies the order of searching directories for executable code, as no conflicts would exist and thus there would be no need to update such an environment variable that specifies directory search order. Thus, the Examiner

must be using Appellants' own specification as the motivation for modifying the teachings of Forbes in accordance with the claimed invention, which is improper hindsight analysis. Thus, in addition to the Examiner failing to properly establish a prima facie showing of obviousness, the Examiner's subjective reasoning as to why such claimed step would have been obvious is itself shown to be flawed.

Quite simply, Forbes's code store data structure is very different from the claimed path sequence of an environment variable as the code store data structure does not provide any directory search order information, as there is no reason to search multiple directories in locating executable code as the directory location for a program is explicitly specified by Forbes in this code store data structure.

A.2. Claims 24, 27 and 30

Appellants urge that there is no teaching or suggestion in the cited Forbes reference of "determining, by the environment variable manager, if any occurring event a), b) or c) *causes a modification to an affected path sequence of any presently active environment variable*, the path sequence specifying an order for searching directories for locating executable code within the data processing system". In rejecting this aspect of Claim 24, the Examiner paints a broad-brushed approach to the teachings of Forbes, and states that Forbes teaches "during the updating and uninstallation process, package manager that detects all change/modification in the operating environment and *take the appropriate action*" (emphasis added). Appellants urge that an assertion of "take appropriate action" does not establish a specific teaching or suggestion of the specifically recited claimed feature of determining if an event *caused modification to an affected path sequence*, the path sequence specifying an order for searching directories for locating executable code. For similar reasons to those described above with respect to Claim 1, since Forbes teaches an explicit directory reference of where programs are stored (which is maintained in the code store data structure), there would be no reason to detect modifications to a path sequence (which specifies a directory search order), as directories are not searched to located executable files (and hence the search order for searching directories is not important or otherwise needed) because their directory location is explicitly maintained. Thus, as there are claimed features not taught or suggested by the cited reference, it is shown that Claim 24 has been erroneously rejected.

A.3. Claims 26, 29 and 32

Appellants urge that there is no teaching or suggestion by Forbes of “*determining if a directory, specified by a path sequence of any environment variable, is deleted*”, wherein the path sequence specifies an order for searching directories for locating executable code within the data processing system”, or “enabling at least one of the following: a) an automatic *deletion of the directory from the path sequence of the environment variable*; and b) a display of an interface to inform the user to delete the directory from the path sequence of any affected environment variables”. As can be seen, Claim 26 is directed to managing of entries of a path sequence of an environment variable, and in particular either automatically deletes the directory from such path sequence, or informs a user to delete the directory from the path sequence. The two actions (automatic or manual deletion of a directory from the path sequence) are based upon a determination that a particular directory (one that is specified by such path sequence) has been deleted. Importantly, it is not merely the determination of deletion of a directory, but rather the determination is with respect to a particular type of directory that has been deleted – a directory that is specified by a path sequence of an environment variable (where such path sequence specifies an order for searching directories for locating executable code within a data processing system). The Examiner rejects Claim 26 for the identical reason as Claim 24 (per paragraph 7 in the Office Action dated 9/29/2004, it is stated “Claims 26, 27, 29, 30 and 32 are rejected on the same ground as stated in claim 24 above”). The Examiner has therefore failed to establish a prima facie showing of obviousness with respect to Claim 26, as the Examiner has failed to establish, or even allege, a teaching or suggestion of “*determining if a directory, specified by a path sequence of any environment variable, is deleted*”, wherein the path sequence specifies an order for searching directories for locating executable code within the data processing system”, or “enabling at least one of the following: a) an automatic *deletion of the directory from the path sequence of the environment variable*; and b) a display of an interface to inform the user to delete the directory from the path sequence of any affected environment variables” (as neither of these steps is recited in Claim 24, and thus the Examiner’s reliance on the reasoning given with respect to Claim 24 is inadequate for rejecting Claim 26). Thus, Claim 26 is shown to have been erroneously rejected. In addition, as a prima facie case of obviousness has not been established, the burden has not shifted to Appellants to rebut an (improper) obviousness assertion. *In re Oetiker, supra*.

B. GROUND OF REJECTION 2 (Claims 7, 9, 16, 23, 25, 28 and 31)**B.1. Claims 7, 16 and 23**

It is error to reconstruct the patentee's claimed invention from the prior art by using the patentee's claims as a "blueprint". When prior art references require selective combination to render obvious a subsequent invention, there must be some reason for the combination other than the hindsight obtained from the invention itself. *Interconnect Planning Corp. v. Feil*, 774 F.2d 1132, 227 USPQ 543 (Fed. Cir. 1985). In rejecting Claim 7, the Examiner states that it would have been obvious to incorporate the teachings of Hove with Forbes to provide a manual option to allow the user to perform such operation during software installation or upgrade. Appellants urge that Forbes expressly states a desire to automate his procedures *to avoid any user involvement or intervention* (Forbes paragraph 0019). Thus, the only motivation for making this change to the teachings of Forbes with those of Hove must be coming from Appellants' own patent specification, which is improper hindsight analysis. Therefore, Claim 7 is shown to have been erroneously rejected.

Appellants further show error in the rejection of Claim 7 for similar reasons to those given above with respect to Claim 1 (of which Claim 7 ultimately depends upon).

B.2. Claim 9

With respect to Claim 9, such claim recites steps of detecting that an environment variable has been modified during the installing step; responsive to the detecting step, determining if duplicate files exist in a path sequence of the modified environment variable; responsive to a determination that duplicate files exist in a path sequence of the environment variable, prompting a user to select a correct one of the duplicate files; and removing all incorrect ones of the duplicate files from the path sequence of the environment variable. As can be seen, this claim is directed to a technique for correcting modifications made to a path sequence of an environment variable, such modifications being made during installation of software. The Examiner states that Forbes discloses detecting and removing the duplicate and incorrect file versions and referencing the appropriate file automatically. Appellants urge that Claim 9 is not merely directed to deletion of duplicate files themselves, but rather is directed to removing incorrect references to duplicate files in a path sequence of an environment variable. None of the cited references teach or suggest any determination that duplicate files exist in a path sequence of

the environment variable, and thus it necessarily follows that neither cited reference teaches or suggests the claimed step of *responsive to a determination that duplicate files exist in a path sequence of the environment variable*, prompting a user to select a correct one of the duplicate files. Thus, the Examiner has failed to properly establish a prima facie showing of obviousness as all the claimed features are not taught or suggested by the cited references. Therefore, Claim 9 is shown to have been erroneously rejected.

Appellants further urge that the Examiner is using improper hindsight with respect to the claimed step of "prompting a user to select a correct one of the duplicate files" for similar reasons to those given above with respect to Claim 7.

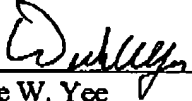
B.3. Claims 25, 28 and 31

With respect to Claim 25, such claim recites (i) automatically invoking an environment variable manager whenever a path sequence for a presently active environment variable is modified in the data processing system, wherein the path sequence specifies an order for searching directories for locating executable code within the data processing system; (ii) determining, by the environment variable manager, if duplicate files exist in the directories specified by the path sequence of the environment variable; and (iii) enabling a display of each environment variable determined to have duplicate files in the directories specified by the path sequence to a user for correction. The Examiner reject Claim 25 for the identical reason as Claim 9 (per paragraph 12 in the Office Action dated 9/29/2004, it is stated "Claims 25, 28 and 31 are rejected on the same ground as stated in claim 9 above"). The Examiner has therefore failed to establish a prima facie showing of obviousness with respect to Claim 25, as the Examiner has failed to establish, or even allege, a teaching or suggestion of (i) automatically invoking an environment variable manager (whenever a path sequence for a presently active environment variable is modified in the data processing system, wherein the path sequence specifies an order for searching directories for locating executable code within the data processing system); (ii) determining, by the environment variable manager, if duplicate files exist in the directories specified by the path sequence of the environment variable; and (iii) enabling a display of each environment variable determined to have duplicate files in the directories specified by the path sequence to a user for correction (none of these steps are recited in Claim 9, and thus the Examiner's reliance on the reasoning given with respect to Claim 9 is inadequate for rejecting Claim 25). Quite simply, Claim 9 is directed

to correcting modifications made to an environment variable during software installation, whereas Claim 25 is directed to managing environment variables by determining whether duplicate files exist in the directories specified by the path sequence of the environment variable, and thus Claims 9 and 25 are not co-extensive. Thus, it is shown that Claim 25 has been erroneously rejected, as a proper prima facie case of obviousness has not been established. In addition, as a prima facie case of obviousness has not been established, the burden has not shifted to Appellants to rebut an (improper) obviousness assertion. *In re Oetiker, supra*.

Appellants further urge that the Examiner is using improper hindsight with respect to the claimed step of "enabling a display of each environment variable determined to have duplicate files in the directories specified by the path sequence to a user for correction" for similar reasons to those given above with respect to Claim 7.

In conclusion, Appellants have shown numerous errors in the Examiner's final rejection of Claims 1, 6, 7, 9, 10, 15-17 and 22-32, and respectfully requests that the rejection of all such claims be reversed by the Board.



Duke W. Yee
Reg. No. 34,285
Wayne P. Bailey
Reg. No. 34,289
YEE & ASSOCIATES, P.C.
PO Box 802333
Dallas, TX 75380
(972) 385-8777
Attorneys for Appellants

CLAIMS APPENDIX

The text of the claims involved in the appeal are:

1. A method for correcting a path sequence of an environment variable in a data processing system, the path sequence specifying an order for searching directories for locating executable code within the data processing system, the method comprising:

monitoring the data processing system for a change effecting the path sequence of the environment variable, wherein the environment variable is enabled and being used by the data processing system to specify the order for searching the directories within the data processing system;

responsive to detection of the change effecting the path sequence of the environment variable, determining whether any duplicate files exist in any of the directories identified by the path sequence; and

responsive to determining that duplicate files do exist, altering the path sequence of the environment variable to ensure that a proper file is found and used when selected by one of a user and a running application program.

6. The method as recited in claim 1, wherein the step of altering the path sequence of the environment variable comprises removing references to all but one of duplicate files in the path sequence of the environment variable.

7. The method as recited in claim 6, wherein the all but one duplicate file is selected by a user.

9. A method for correcting modifications that have been made to an environment variable during installation of software in a data processing system, the method comprising:

installing the software on the data processing system for subsequent execution by the data processing system;

detecting that an environment variable has been modified during the installing step;

responsive to the detecting step, determining if duplicate files exist in a path sequence of the modified environment variable;

responsive to a determination that duplicate files exist in a path sequence of the environment variable, prompting a user to select a correct one of the duplicate files; and

removing all incorrect ones of the duplicate files from the path sequence of the environment variable.

10. A computer program product in a computer readable media for use in a data processing system for correcting a path sequence of an environment variable in the data processing system, the path sequence specifying an order for searching directories for locating executable code within the data processing system, the computer program product comprising:

first instructions for monitoring the data processing system for a change effecting the path sequence of the environment variable, wherein the environment variable is enabled and being used by the data processing system to specify the order for searching the directories within the data processing system;

second instructions, responsive to detection of the change effecting the path sequence of the environment variable, for determining whether any duplicate files exist in any of the directories identified by the path sequence; and

third instructions, responsive to determining that duplicate files do exist, for altering the path sequence of the environment variable to ensure that a proper file is found and used when selected by one of a user and a running application program.

15. The computer program product as recited in claim 10, wherein the step of altering the path sequence of the environment variable comprises removing references to all but one of duplicate files in the path sequence of the environment variable.

16. The computer program product as recited in claim 15, wherein the all but one duplicate file is selected by a user.

17. A system for correcting a path sequence of an environment variable in a data processing system, the path sequence specifying an order for searching directories for locating executable code within the data processing system, the system comprising:

first means for monitoring the data processing system for a change effecting the path sequence of the environment variable, wherein the environment variable is enabled and being used by the data processing system to specify the order for searching the directories within the data processing system;

second means, responsive to detection of the change effecting the path sequence of the environment variable, for determining whether any duplicate files exist in any of the directories identified by the path sequence; and

third means, responsive to determining that duplicate files do exist, for altering the path sequence of the environment variable to ensure that a proper file is found and used when selected by one of a user and a running application program.

22. The system as recited in claim 17, wherein the step of altering the path sequence of the environment variable comprises removing references to all but one of duplicate files in the path sequence of the environment variable.

23. The system as recited in claim 22, wherein the all but one duplicate file is selected by a user.

24. A method for managing environment variables in a data processing system, comprising data processing system implemented steps of:

automatically invoking an environment variable manager upon occurrence of at least one of occurring events: a) a directory is deleted; b) a product is uninstalled on the data processing system; and c) a given environment variable is manually modified by a user;

determining, by the environment variable manager, if any occurring event a), b) or c) causes a modification to an affected path sequence of any presently active environment variable, the path sequence specifying an order for searching directories for locating executable code within the data processing system; and

automatically correcting the affected path sequence if it is determined that the occurring event causes the modification.

25. A method for managing environment variables in a data processing system, comprising data processing system implemented steps of:

automatically invoking an environment variable manager whenever a path sequence for a presently active environment variable is modified in the data processing system, wherein the path sequence specifies an order for searching directories for locating executable code within the data processing system;

determining, by the environment variable manager, if duplicate files exist in the directories specified by the path sequence of the environment variable; and

enabling a display of each environment variable determined to have duplicate files in the directories specified by the path sequence to a user for correction.

26. A method for managing environment variables in a data processing system, comprising data processing system implemented steps of:

determining if a directory, specified by a path sequence of any environment variable, is deleted, wherein the path sequence specifies an order for searching directories for locating executable code within the data processing system; and

enabling at least one of the following: a) an automatic deletion of the directory from the path sequence of the environment variable; and b) a display of an interface to inform the user to delete the directory from the path sequence of any affected environment variables.

27. A data processing system having means for managing environment variables, comprising:
means for automatically invoking an environment variable manager upon occurrence of at least one of occurring events: a) a directory is deleted; b) a product is uninstalled on the data processing system; and c) a given environment variable is manually modified by a user;

means for determining, by the environment variable manager, if any occurring event a), b) or c) causes a modification to an affected path sequence of any presently active environment variable, the path sequence specifying an order for searching directories for locating executable code within the data processing system; and

means for automatically correcting the affected path sequence if it is determined that the occurring event causes the modification.

28. A data processing system having means for managing environment variables, comprising:
means for automatically invoking an environment variable manager whenever a path sequence for a presently active environment variable is modified in the data processing system, wherein the path sequence specifies an order for searching directories for locating executable code within the data processing system;

means for determining, by the environment variable manager, if duplicate files exist in the directories specified by the path sequence of the environment variable; and

means for enabling a display of each environment variable determined to have duplicate files in the directories specified by the path sequence to a user for correction.

29. A data processing system having means for managing environment variables in a data processing system, comprising:

means for determining if a directory, specified by a path sequence of any environment variable, is deleted, wherein the path sequence specifies an order for searching directories for locating executable code within the data processing system; and

means for enabling at least one of a) an automatic deletion of the directory from the path sequence of the environment variable; and b) a display of an interface to inform the user to delete the directory from the path sequence of any affected environment variables.

30. A computer program on a computer readable medium having program code means for managing environment variables in a data processing system, comprising:

instruction means for automatically invoking an environment variable manager upon occurrence of at least one of occurring events: a) a directory is deleted; b) a product is uninstalled on the data processing system; and c) a given environment variable is manually modified by a user;

instruction means for determining, by the environment variable manager, if any occurring event a), b) or c) causes a modification to an affected path sequence of any presently active environment variable, the path sequence specifying an order for searching directories for locating executable code within the data processing system; and

instruction means for automatically correcting the affected path sequence if it is determined that the occurring event causes the modification.

31. A computer program on a computer readable medium having program code means for managing environment variables in a data processing system, comprising:

instruction means for automatically invoking an environment variable manager whenever a path sequence for a presently active environment variable is modified in the data processing system, wherein the path sequence specifies an order for searching directories for locating executable code within the data processing system;

instruction means for determining, by the environment variable manager, if duplicate files exist in the directories specified by the path sequence of the environment variable; and

instruction means for enabling a display of each environment variable determined to have duplicate files in the directories specified by the path sequence to a user for correction.

32. A computer program on a computer readable medium having program code means for managing environment variables in a data processing system, comprising:

instruction means for determining if a directory, specified by a path sequence of any environment variable, is deleted, wherein the path sequence specifies an order for searching directories for locating executable code within the data processing system; and

instruction means for enabling at least one of the following a) an automatic deletion of the directory from the path sequence of the environment variable; and b) a display of an interface to inform the user to delete the directory from the path sequence of any affected environment variables.

EVIDENCE APPENDIX

There is no evidence to be presented.

RELATED PROCEEDINGS APPENDIX

There are no related proceedings.